Attorney's Docket No.: 10559-566001 / P12728

APPLICATION

FOR

UNITED STATES LETTERS PATENT

TITLE: METHOD FOR EXTENDING THE LOCAL MEMORY ADDRESS SPACE OF A PROCESSOR

APPLICANT: RAVI KOLAGOTLA AND JUAN G. REVILLA

CERTIFICATE OF MAILING BY EXPRESS MAIL

Express Mail Label No. EL688323092US

I hereby certify under 37 CFR §1.10 that this correspondence is being deposited with the United States Postal Service as Express Mail Post Office to Addressee with sufficient postage on the date indicated below and is addressed to the Commissioner for Patents, Washington, D.C. 20231.

12.28.01
Date of Deposit

Gabriel Lewis
Signature

Typed or Printed Name of Person Signing Certificate

METHOD FOR EXTENDING THE LOCAL MEMORY ADDRESS SPACE OF A PROCESSOR

BACKGROUND

[0001] Memory in a computer system may be arranged in a memory hierarchy including memory devices of different speeds and sizes. The type and size of a memory device and its proximity to the processor core are factors in the speed of the memory device. Generally smaller hardware is faster, and memory devices closest to the processor core are accessed fastest. Since fast memory may be expensive and space near the processor core limited, a memory hierarchy may be organized into several levels, each smaller, faster, and more expensive per byte than the next level. The goal of such a memory hierarchy is to provide a memory system with a cost almost as low as the cheapest level of memory and speed almost as fast as the fastest level of memory.

[0002] Many processors use memory caches to store copies of the most used data and instructions in order to improve access speed and overall processing speed. A memory cache, also referred to as cache store or RAM (Random Access Memory) cache, is a portion of memory which may be made of high-speed static RAM (SRAM) instead of the slower dynamic

RAM (DRAM) used for main memory. Memory caches may be included at the highest level of memory and on the same integrated circuit (IC) as the processor. Such internal memory caches are also referred to as local or Level 1 (L1) caches.

[0003] The contents of the L1 cache may change depending on the task being performed by the processor. If the processor tries to access data that is not in the cache, a cache miss occurs, and the data must be retrieved from a lower level of memory. Cache misses involve a performance penalty, which includes the clock cycle in which the miss occurs and the number of cycles spent recovering the requested data from memory. Accordingly, it may be desirable to provide a local addressable memory, e.g., an L1 SRAM, to store data and instructions in the processor core to improve access speed and reduce cache miss penalties.

BRIEF DESCRIPTION OF THE DRAWINGS

[0004] Figure 1 is a block diagram of a processor according to an embodiment.

[0005] Figures 2A-2C illustrate a flowchart describing a memory access operation according to an embodiment.

[0006] Figure 3 is a block diagram of a system including a processor according to an embodiment.

DETAILED DESCRIPTION

[0007] Figure 1 illustrates a system 100 according to an embodiment. The system includes a processor 102 with a processor core 105 which interprets and executes software instructions. The processor core 105 may access data from an external memory 110, e.g., a Level 2 (L2) or main memory, via a system interface bus (SBI) 115.

[0008] The processor 102 may be, for example, a microcontroller or a digital signal processor (DSP), which are typically used for controller-oriented applications and numerically-intensive digital signal processing, respectively. The processor 102 may have a hybrid microcontroller/DSP architecture which is able to handle applications which have both DSP- and microcontroller-based components. Such a processor may be used in, for example, a cellular phone which has a workload with a large DSP component for performing the processing required for the base-band channel and the speech coders, as well as a control-oriented application for managing aspects of the user interface and communication protocol stacks.

[0009] The processor core 105 may include a local, or Level 1 (L1), memory level. The L1 memory level may include an L1 memory cache 115 which store copies of the most used data for fast retrieval by an execution unit 120. The contents in the L1 cache 115 may change depending on the tasks being performed by the processor 102.

[0010] The instructions and data in the L1 cache 115 may be stored separately in an L1 instruction cache (I-cache) 125 and an L1 data cache (D-cache) 130, respectively, but may share a common memory at the second and further levels of the system (L2 and lower). The separation of the instruction and data streams may enable the processor core 105 to simultaneously fetch instructions and load/store data without collisions.

[0011] The execution unit 120 may request access to memory. A memory controller 135 may check the address of the requested memory location and send the access to the L1 cache 115. If the L1 cache 115 has a copy of the requested information (cache hit), the L1 cache returns the requested information. A cache miss occurs when the processor core 105 tries to access data that is not in the L1 cache. In the event of a cache miss, the cache attempts to retrieve the requested data from the external memory 140. The retrieved data is transferred to the L1 cache from the

external memory 140 via the SBI 110. A cache miss involves a penalty which includes the clock cycle in which the miss occurred and the additional clock cycles to service the miss.

[0012] The processor core 105 may include local (L1) addressable memory, e.g., an L1 SRAM (Static Random Access Memory) 145. The instructions and data in the L1 memory may be separated into an instruction SRAM (I-SRAM) 150 and a data SRAM (D-SRAM) 155, but may share a common memory at the second and further levels of the system (L2 and lower). Unlike the L1 caches, the L1 SRAMs are "real" memory and will return requested information if it exists. Thus, accesses to L1 SRAM may not entail cache misses and the associated penalties. The L1 SRAM 145 may be programmed with instructions and data used in, for example, DSP-critical applications, such as fast Fourier processing (FFP), correlation, and multiply-accumulate (MAC) operations.

[0013] Some of the system memory may be mapped in the L1 memory address space and some memory may be mapped in the L2 and lower memory address spaces. Every region in memory may be described in a page. A page is a fixed-sized block of memory and the basic unit of virtual memory. The processor 102 may support different page sizes, e.g., 1 kB,

4 kB, 1 MB, and 4 MB. Pages may have properties, e.g., cacheability and protection properties. These properties may be identified by page descriptors such as Cacheability Protection Look-aside Buffer (CPLB) descriptors and Translation Look-aside Buffer (TLB) descriptors. One such descriptor may be a local memory descriptor, e.g., an "LI SRAM" bit, which may be defined on a page-by-page basis and identify a page as being in the L1 logical address space or not, e.g., by being set to "1" or "0", respectively.

[0014] Figures 2A-2C illustrate a flowchart describing a memory access operation 200 according to an embodiment. The local memory controller 135 may handle memory access requests from the execution unit 120. When the execution unit requests an access to memory (block 202), the local memory controller 135 may examine the upper bits of the memory address (block 204) to determine the page in which the address resides (block 206). The local memory controller may check the L1 SRAM bit in the page descriptor to determine if the page is in the L1 memory space (block 208).

[0015] If the L1 SRAM bit is "1", indicating that the page is in the L1 address space, the local memory controller 135 sends the access to the L1 SRAM 115 (block

212). If the address exists in the L1 SRAM, the L1 SRAM will return the requested data (block 214).

[0016] The execution unit 120 may request access to non-existent memory. This may occur due to mistakes in the program and in instances when the program wanders outside of the enabled L1 SRAM memory address space. If the access is to non-existent L1 SRAM memory, the local memory controller 135 may trigger an illegal-access violation exception (block 216). The execution flow may then be interrupted in order for the processor 102to handle the exception.

[0017] If the L1 SRAM bit is set to "0", indicating that the address is not in the L1 address space, the local memory controller 135 may send the access to the L1 cache 115 (block 218). If a copy of the data exists in the L1 cache, the cache will return the requested data (block 220). In the event of a cache miss, the cache may perform an external memory access (block 222).

[0018] The local memory descriptor enables efficient access to local memory when local memory exists in parallel with local cache, making it unnecessary to send the access to both the L1 cache and L1 SRAM simultaneously. Since local memory requests are routed immediately to the L1 SRAM and the L1 cache does not receive such requests, the local

memory controller 135 can quickly determine if an external access needs to be performed. Also, the local memory controller can prevent external memory accesses from being performed (with the associated penalties) for known non-existent memory.

[0019] The local memory descriptors and other page descriptors may be stored in a descriptor buffer. The buffer may hold a limited number of descriptor entries. Thus, using a larger page size may enable more memory to be mapped efficiently. For example, a 64 kB L1 SRAM may store sixteen 4 kB pages. Sixteen local memory descriptor entries would be needed to identify these sixteen pages. Alternatively, the entire L1 memory address space could be contained in one 1 MB page, requiring only one local memory descriptor. As long as the processor 102 accessed only the enabled portion, or separate enabled sub-portions, of the address space in the page, no illegal-accesses violation exceptions would be triggered.

[0020] The processor 102 may be implemented in a variety of systems including general purpose computing systems, digital processing systems, laptop computers, personal digital assistants (PDAs) and cellular phones. In such a system, the processor may be coupled to a memory device, such as a Flash memory device or a static random access

memory (SRAM), which stores an operating system or other software applications.

[0021] Such a processor 102 may be used in video camcorders, teleconferencing, PC video cards, and High-Definition Television (HDTV). In addition, the processor 102 may be used in connection with other technologies utilizing digital signal processing such as voice processing used in mobile telephony, speech recognition, and other applications.

[0022] For example, Figure 3 illustrates a mobile video device 300 including a processor 102 according to an embodiment. The mobile video device 300 may be a hand-held device which displays video images produced from an encoded video signal received from an antenna 302 or a digital video storage medium 304, e.g., a digital video disc (DVD) or a memory card. The processor 102 may communicate with an L2 SRAM 306, which may store instructions and data for the processor operations, and other devices, for example, a USB (Universal Serial Bus) interface 308.

[0023] The processor 102 may perform various operations on the encoded video signal, including, for example, analog-to-digital conversion, demodulation, filtering, data recovery, and decoding. The processor 100 may decode the compressed digital video signal according to one of various

digital video compression standards such as the MPEG-family of standards and the H.263 standard. The decoded video signal may then be input to a display driver 310 to produce the video image on a display 312.

[0024] A number of embodiments have been described. Nevertheless, it will be understood that various modifications may be made without departing from the spirit and scope of the invention. For example, blocks in the flowchart may be skipped or performed out of order and still provide desirable results. Accordingly, other embodiments are within the scope of the following claims.